EXHIBIT 203

Dynamic Revenue Sharing (DRS) V2 Proposal

mirrokni@,renatoppl@
in collaboration with whlin@, jimgiles@, nitish@, mpal@, gagangoel@

Background and Summary

Dynamic revenue sharing (DRS) for DRX is a new way of changing revenue share per auction and as a result, increase the fill-rate and total revenue of publishers in DRX. Several variants have been proposed for this purpose. Here, after comparing with several variants, we propose a "revised 2sided DRS" (the idea is summarized below) as the DRS V2. This revenue sharing mechanism encourages seller-friendly bidding from buyers by encouraging them to declare two bids from their advertisers. More notably, compared to DRS V1 and other variants, it

- brings more revenue lift for publishers (even while charging the reserve price instead of first price in the dynamic region),
- keeps AdX's margin at 20% (& thus it brings (much) more profit lift to AdX),
- more explicitly takes into account incentive issues from both buyer and seller sides across auctions, and results in much less opportunity to game the auction by bid shading (compared to V1 and other non-truthful variants),
- strictly increases ROI for seller-friendly buyers (e.g., GDN) by increasing their utility per each auction,
- treats seller-friendly buyers (e.g., GDN) the same as DRS V1 (in terms of pricing & allocation),
- finally, it encourages declaring two bids for the exchanges.

In this document, we present a detailed comparison to other DRS methods and show the above points. Most notably, our experiments show that this algorithm not only brings more revenue for AdX and publishers, but also it increases RPM for publishers a bit. At the same time, its impact on the advertiser RPM and publisher RPM is not significant. Here, we present this DRS method, report its comparison to V1 and other truthful and non-truthful variants.

Compared to DRS V1, we get considerable more profit (at least 53% profit lift with a conservative estimate). Finally, we note that by implementing DRS V2, we can apply a less tight throttling probability for DRS V2 compared to DRS V1, and therefore we can get much more revenue (>55% more revenue lift, and > 169% more profit lift).

Links to previous proposals: DRS V1, Enhanced DRS, Two-sided DRS, Truthful DRS

Description of the Revenue Sharing Scheme

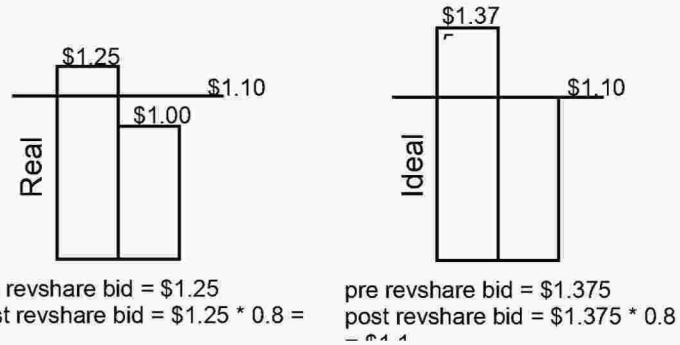
The main goal with DRS is to be able to clear impressions for which the highest bid is very close to the reserve. Currently sellside revshare[1] is fixed at 80% and the computation of revenue (how

[1] there is also buyside revshare. The bid submitted by the advertiser is multiplied by buyside revshare (which can be smaller or larger than 1.0 due to Bernanke) and then multiplied by sellside revshare

much we charge the advertiser) and payout (how much we pay the publisher) is done as follows:

- post_revshare_bid = bid * revshare (which map bids to publisher space)
- if no post_revshare_bid above the publisher reserve, the auction doesn't clear.
- o.w. payout = max(reserve, second highest post_revshare_bid)
  revenue = payout / revshare

So a bid of $1.25 won't win an impression with reserve $1.10, since first we take 80% of sellside revshare, so the post-revshare bid is $1 and the impression doesn't clear. See example below



pre revshare bid = $1.25
post revshare bid = $1.25 * 0.8 =

pre revshare bid = $1.375
post revshare bid = $1.375 * 0.8

The two-sided dynamic revshare proposal is to clear impressions that would clear if it were not for revshare, i.e., impressions for which **bid * revshare < reserve < bid**. We say say such impressions are in the ***dynamic region***.

In the two-sided design, whenever an impression is in the dynamic regions, we clear it and charge:

- revenue = max(reserve, second highest bid)
- payout = reserve

In this query, the publisher has up to 100% of the revshare. So in order to recover Google's 20% cut on this transaction we have for each buyer and for each publisher a **debt account** in which we store the amounts we should have been paid in such queries:

- debt [buyer] += reserve / revshare - revenue
- debt [publisher] += payout - revenue * revshare

We attempt to collect debt in a later query for which bid * revshare > reserve, which we call the **non-dynamic region**. In such query we do as follows:

- $payout^0$ = max(reserve, second highest post_revshare_bid)
- $revenue^0$ = $payout^0$ / revshare
- revenue = $revenue^0$ + buyer_collection

(currently constant at 80%). In this doc, we assume buyside revshare is always 1.0 (or equivalently, when we say pre-revshare, we actually mean, post-buyside-pre-sellside revshare.